JSON WITH PHP

This tutorial will teach you how to encode and decode JSON objects using PHP programming language. Let's start with preparing environment to start our programming with PHP for JSON.

Environment

As of PHP 5.2.0, the JSON extension is bundled and compiled into PHP by default.

JSON Functions

Function	Libraries
json_encode	Returns the JSON representation of a value
json_decode	Decodes a JSON string
json_last_error	Returns the last error occurred

Encoding JSON in PHP (json_encode)

PHP json_encode() function is used for encoding JSON in PHP. This function returns the JSON representation of a value on success or FALSE on failure.

Syntax:

```
string json_encode ( $value [, $options = 0 ] )
```

Parameters:

- **value**: The value being encoded. This function only works with UTF-8 encoded data.
- **options**: This optional value is a bitmask consisting of JSON_HEX_QUOT, JSON_HEX_TAG, JSON_HEX_AMP, JSON_HEX_APOS, JSON_NUMERIC_CHECK,JSON_PRETTY_PRINT, JSON_UNESCAPED_SLASHES, JSON_FORCE_OBJECT

Example

The following example shows how to convert an arrays into JSON with PHP:

```
<?php
   $arr = array('a' => 1, 'b' => 2, 'c' => 3, 'd' => 4, 'e' => 5);
   echo json_encode($arr);
?>
```

While executing, this will produce following result:

```
{"a":1,"b":2,"c":3,"d":4,"e":5}
```

The following example shows how PHP objects can be converted into JSON:

```php
<?php
   class Emp {
       public $name = "";
       public $hobbies  = "";
       public $birthdate = "";
   }
   $e = new Emp();
   $e->name = "sachin";
   $e->hobbies  = "sports";
   $e->birthdate = date('m/d/Y h:i:s a', "8/5/1974 12:20:03 p");

   echo json_encode($e);
?>
```

While executing, this will produce following result:

```
{"name":"sachin","hobbies":"sports","birthdate":"12\/31\/1969 05:00:08 pm"}
```

Decoding JSON in PHP (json_decode)

PHP json_decode() function is used for decoding JSON in PHP. This function returns the value decoded from json to appropriate PHP type.

Syntax:

```
mixed json_decode ($json [,$assoc = false [, $depth = 512 [, $options = 0 ]]])
```

Paramaters:

- **json_string**: It is encoded string which must be UTF-8 encoded data
- **assoc**: It is a boolean type parameter, when set to TRUE, returned objects will be converted into associative arrays.
- **depth**: It is an integer type parameter which specifies recursion depth
- **options**: It is an integer type bitmask of JSON decode, JSON_BIGINT_AS_STRING is supported.

Example

The following example shows how PHP can be used to decode JSON objects:

```php
<?php
   $json = '{"a":1,"b":2,"c":3,"d":4,"e":5}';

   var_dump(json_decode($json));
   var_dump(json_decode($json, true));
?>
```

While executing, this will produce following result:

```
object(stdClass)#1 (5) {
    ["a"] => int(1)
    ["b"] => int(2)
    ["c"] => int(3)
    ["d"] => int(4)
    ["e"] => int(5)
}

array(5) {
```

```
    ["a"] => int(1)
    ["b"] => int(2)
    ["c"] => int(3)
    ["d"] => int(4)
    ["e"] => int(5)
}
```